11. Rohit

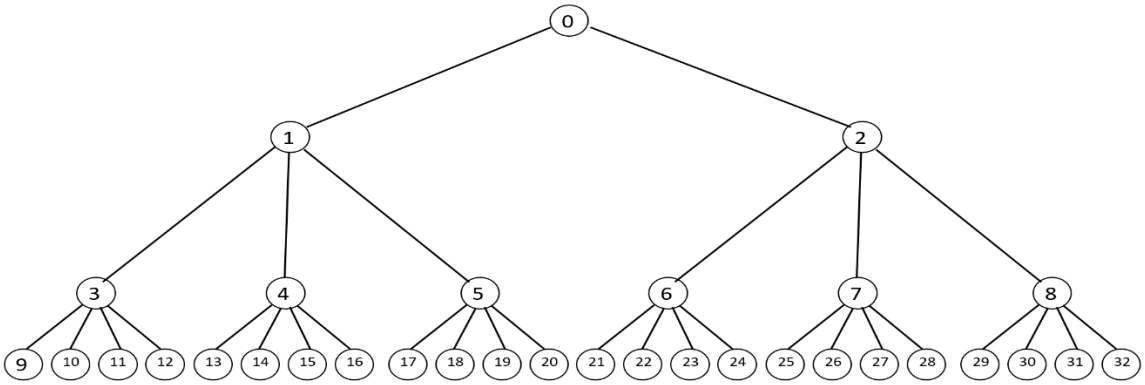
Program Name: Rohit.java Input File: rohit.dat

Rohit just learned about binary trees and is so excited that he has decided to invent a new kind of tree data structure!

Binary trees are trees where each node has 2 or fewer children. However, trees can have any number of children, even a variable number of children. Each level of Rohit's trees may have any number of children. There will always be a single node root for his trees.

He would like your help implementing the **getChildren** method of this new tree. To prove that your implementation works, Rohit has asked that given the number of children at each level of the tree beyond the root, you build a tree filled incrementally by level. Then using that tree test your **getChildren** method.

For example, if Rohit gives you "2, 3, 4" then this corresponds to a tree where the root has 2 children, each of those 2 children have 3 children, and each of those have 4 children. The resulting tree would look like this:



If Rohit calls getChildren(4), the result would be "13 14 15 16", all of the children of the node in position 4. If he asked for children of "0" the answer would be "1 2". If he asks for the children of a node that has no children, like any of the nodes on the last level, such as 19, then the answer would be "NO CHILDREN". If Rohit asks for the children of a node that isn't in the tree, like 33, then answer would be "NOT IN TREE".

Input: The first integer is the number of data sets to follow. Each data set will have two lines, the first line is a list of integers representing the number of children at each level of the tree. The second integer is the node being used in the getChildren method call.

Output: The list of children of the node in question, "NO CHILDREN", or "NOT IN TREE".

Sample input:

```
4
2 3 4
4
5 2 2
23
2 2 2 2 2 2 2
119
1 1 1 1 1
12
```

Sample output:

```
13 14 15 16
NO CHILDREN
239 240
NOT IN TREE
```